An Object-Oriented Scheduling Architecture for Managing the Data Relay Satellite Requests

Amedeo Cesta
IP-CNR
National Research Council
Viale Marx 15
I-00137 Rome, Italy
amedeo@pscs2.irmkant.rm.cnr.it

Paolo Bazzica and **Gianni Casonato**
Corso di Laurea
Ingegneria Informatica
Università di Roma "La Sapienza"
I-00198 Rome, Italy
{bazzica, casonato}@dis.uniroma1.it

Abstract

In this paper an automated system is described for managing the daily activities of the DRS satellite system. In the system the object-oriented and artificial intelligence methodologies have been jointly used to develop a comprehensive approach to the problem. Particular attention has been given to the representation of the scheduling domain, the dynamic maintenance of a solution, and to the interaction with different types of users. The system offers flexible services and its performance is acceptable for the operative environment.

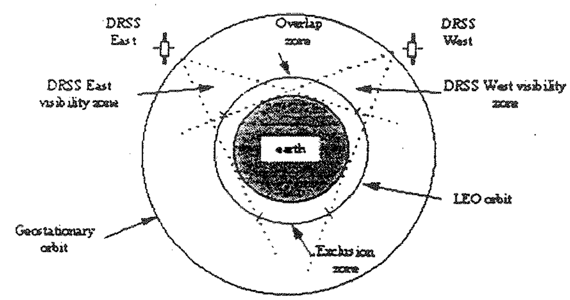

Figure 1: The Data Relay Satellite System

Problem Description

The Data Relay Satellite (DRS) System is an European Space Agency program aimed at providing a data relay service between Low Earth Orbiting (LEO) satellites and their ground terminals (ESA 1989; 1990). Actually this program is in the last step of development, and it will be operative within 1999.

The DRS infrastructure (see Figure 1) consists of a constellation of two satellites and in a set of ground stations that allow:

- almost total coverage area;
- strong reduction of the LEO's ground terminal network;
- reduction in data-distribution problems;
- reduction of required on-board data storage capacity for LEO satellites.

The scheduling problem of DRS consists in the production of a mission plan, that allow the clients to utilize the transmission services. An high number of access requests is expected, so that their temporal extension exceeds the total transmission time available, introducing conflicts that have to be solved following some quality objectives.

Given the technical characteristics of the DRS system, the crucial aspect in the production of the plan is the management of the link between the DRS and the LEO satellites, while the links between DRS and ground stations are less problematic. The first type of link imposes the satisfaction of physical constraints of the DRS's antennas, temporal constraints of the requests, and requirements of priority, commercial value and allocation preference.

To summarize the problem addressed, we use the usual scheduling terminology (French 1982) identifying problem's resources, activities, activities constraints, and optimum criteria. It is worth noting that from now on we speak of requests instead of activities, due to the particular application domain.

Resources. The physical system modeled by the application consists of one out of the two DRSs, composed of two Inter Orbit Links (IOL). Consequently, a request can be allocated if an interval of time can be established in which an antenna of the IOL is available.

Requests and related constraints. All user requests specify a number of desired characteristics which include:

- static priority associated to the request's owner;
- technical requirements: these include the band, speed of transmission and the number of channels required;
- user flexibilities: minimum and maximum time intervals for the duration, the interval of time within which the access must be scheduled (flexibility interval) and the utility function associated with these flexibilities (see Figure 2);

- user preferences: preferred values for the duration and the actual time of access (see functions f_1 and f_2 in Figure 2).

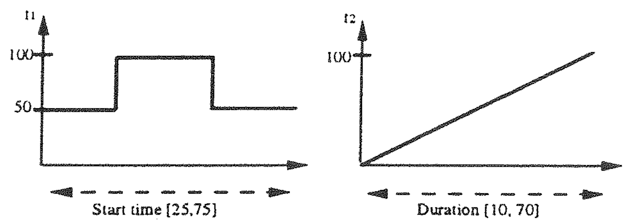

Figure 2: Temporal flexibility and preference functions

Optimization. The goals which need to be satisfied in generating the Detailed Assignment Plan (DAP) are:

- scheduling of as many access requests as possible;
- satisfaction of as many user preferences as possible;
- priority for preferences of requests having a higher 'relevance' coefficient.

The goals are potentially conflicting: an optimization in resource use required to satisfy the first goal would imply taking full advantage of user specified flexibilities but in doing so, the preference (or utility) function given by the users may not be satisfied. The other two goals are in turn partially contrasting, since maximizing user preferences does not necessarily coincide with satisfying the requests of preferred users.

According to the technical documentation, the production of the DAP is supposed to follow an iterative process repeated three times, and that involves two types of human operators:

- *Commercial operators* at the Mission Control Center: negotiates with the clients the sale of the free transmission spaces, and inserts in the plan the related activities;
- *Spacecraft engineers* at the Operation Control Center: modifies the plan inserting in some special activities for the maintenance of the system operativity, and requests with a special requirement of urgency.

These two operational profiles follow different and potentially conflicting objectives (maximum satisfaction of requests vs. DRS's resources saving). Those objectives have to be integrated together in an automated scheduling system that supports decision making in this environment.

Scheduling Architecture Design

Artificial Intelligence (AI) techniques provide an approach to a planning and scheduling problem which is based on three fundamental aspects (Figure 3):

- representation of the domain and solution management;
- generation of satisfactory or optimal solutions;
- interaction with the user.

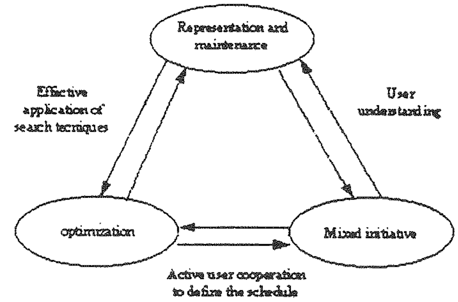

Figure 3: Basic aspects of the AI approach

These three aspects may increase the project complexity for a system that should supply all these characteristics. This problem can be solved using the tools provided from the object-oriented analysis and design techniques.

The representation of the domain has to be dynamic, able to follow the physical changes in the domain and to supply an incremental building of the plan. In the same time has to be symbolic to allow the user high level understanding. The need to product optimal plans, for the high complexity of the scheduling problems, claims the use of heuristics techniques of search. These techniques quickly generate a solution, and then, if necessary, allow the user to directly modify the building process. The ideas of decomposition, abstraction and hierarchy help to individuate the atomic entities of the problem. Each of those entities is in relation (inheritance, aggregation and use) with some other entities at the right abstraction level. We have used object-oriented methodology to realize an architecture for representation and maintenance of a scheduling domain. This architecture, named O-OSCAR (Object-Oriented Scheduling Architecture), can supply all the tools needed for the optimization task and for the interaction needs of the system. At a very abstract level O-OSCAR represents the four basic entities in a scheduling problem (similarly to (Wolf 1994)):

- processors;
- operations;
- decisions;
- constraints.

The complete O-OSCAR architecture is organized in a hierarchical structure, subdivided in two levels: abstract level and concrete level.

The abstract level defines entities with general characteristics and functionalities, common in all scheduling problems. The concrete level contains the concrete entities, derived from the abstract ones, that describe a more specific set of scheduling problems. Following the objective of the maximum applicability of the architecture, the class of satellite scheduling problems (SSP) has been defined. This class contains the DRS problem, and it is constituted by a general job-shop with additional quantitative precedence constraints and allocation preference for the operations. The concrete entities can be instantiated to a real problem (to DRS problem in our case) and by using a transaction model of utilization —based on insert, delete and retrieve actions— they can support the maintenance of a solution as an incremental building strategy.

Figure 4 shows, in Booch notation (Booch 1994), the basic entities in the object-oriented design of both the abstract level and the concrete level.

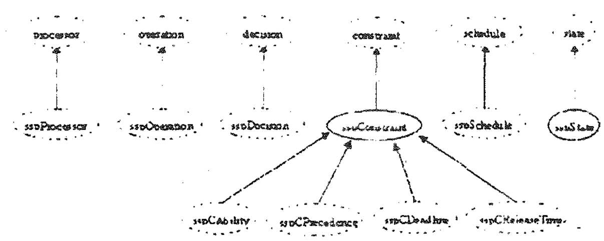

Figure 4: Basic entities in O-OSCAR design

A Scheduler for the DRS

Having implemented a complete architecture able to represent the knowledge about a scheduling domain, we use it to realize a complete activity sequencer for the described satellite system.

The main objective of the project is to keep the user inside the scheduling process. For this reason a study of the scheduling environment had to be performed to define the aspects of the work that mainly need an automatic support. The DRS planning environment supports the two typical users introduced above: the commercial user at the the Mission Control Center and the operative user (spacecraft engineers) at the Operation Control Center.

For each type of users a personalized set of instruments to manipulate the schedule had been defined. This required the definition of two User Profiles. In the DRS sequencer all the aspects common to the two profiles form the kernel of the sequencer, while those typical of a specific profile are isolated in a series of separated modules. The object-oriented design made simpler the adaptation of the system to the needs of different users.

The process of schedule building is at each moment under the user's control. In each planning phase the user has the ability to operate directly on the schedule, if needed. A direct consequence of this user needs is the project choice to symbolically represent the knowledge of the domain. The nature of the defined objects allows the user manipulation at an abstraction level near to the highly symbolic human reasoning. The use of the C++ language, that directly supports the object-oriented decomposition, allows to achieve an high execution efficiency.

The sequencer architecture allows the integration of new heuristics while minimizing the effects on the code. At present the sequencer provides two scheduling heuristics: the first is strictly tied to the problem and allows the construction of the schedule managing the backtracking in a dependency directed way (a greedy heuristics), the second uses a generalized simulated annealing technique to perform the same task. The user has the ability to interactively change the heuristic to be used in each phase of the scheduling process. The availability of a heuristics set claims for the definition of an evaluation method capable to compare the results obtained with different heuristics (or even those due to a same heuristic used with different tuning). In the case of the DRS, the quality evaluation of a schedule is connected to the existence of a set of user-selectable criteria able to evaluate the schedule quality, according to a set of scheduling objectives. The flexible way to do that is by defining a set of functions that associate a quality value to each activity in any particular schedule. The quality of the whole schedule is then defined as the sum of the activities qualities that form it.

In the DRS scheduler three quality functions exist:

- A global quality function that, for each scheduled request, defines a real number directly dependent on (1) the priority of the request, (2) the commercial value of the request (3) the response to the user wishes for the start and the duration of the request.
- A priority function that, for each scheduled request, defines a real value directly dependent on the priority of the request. This function rewards the schedules that accepts the requests with the highest priority.
- A flexibility function that rewards the schedules which better respect the user defined preferences about the start time and the duration of each request accepted.

The choice of define a set of functions instead of just one is due to the need to evaluate the schedule quality from different viewpoints. In general terms, the user has the ability to choose one of these functions to define the scheduling objective. Each time the control of the scheduling process is transferred to the system the schedule will be manipulated so that the selected quality function is maximized. To direct the user towards the best choice with respect to the particular problem involved a comparison criterion between the available heuristics is needed. To this purpose the DRS sequencer evaluates the schedule quality using the Expected Solution Quality (ESQ) theory (Bresina *et al.*

1995). This method has been formulated to statistically evaluate the performances of the scheduling algorithms. By building a set of random-generated solutions a gaussian density quality function is inferred and the standard deviation of the scheduling algorithm used from the mean of this function is used to define a measure of the algorithm quality. The DRS sequencer incorporates an interactive version of the ESQ algorithm able to perform the quality of a partial schedule in each moment of the planning process.

The block diagram of the software system is presented in Figure 5. The modularity of the system, due to the project choice of facing the aspects of knowledge representation, solution optimization and mixed initiative in independent software modules, allows the definition of new heuristics and/or new user profiles without modifying the main structure of the system.

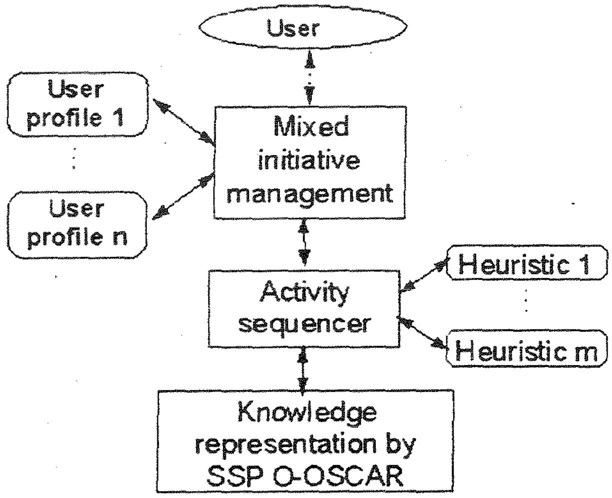

Figure 5: DRS O-OSCAR architecture

The Module named SSP O-OSCAR in Figure 5 is responsible for the representation of the domain and for the maintenance of the current solution. To represent the temporal aspects of the solution the set of dynamic propagation algorithms described in (Cervoni *et al.* 1994; Cesta and Oddi 1996b) are used.

The system can configure its own level of initiative following the user wishes. Its role in the scheduling task can simply consist of supporting the user in his effort by maintaining the information about the schedule and signaling when the user choices conflicts with previous ones. Otherwise the system can try the insertion of a set of requests, specified by the user with the desired heuristic. Finally the system can generate a complete schedule from a set of specified requests allowing a user post-initiative on the schedule.

The choice to implement an interactive system requires the use of a graphical instruments set to interact effectively with the user. This choice imply the risk to compromise the portability of the system due to the incompatibility of the main hardware platforms regarding the graphic interface management. The DRS sequencer give solution to this problem entrusting on the AMULET library, developed in C++ at the Carnegie Mellon University (Myers). This library, allows the creation of a graphical interface abstracted from the hardware by defining a set of instruments like windows, buttons etc. The concepts of object-oriented programming like inheritance, polymorphism and encapsulation are fully implemented allowing the coding of the interface at a very high level of abstraction. The fact that the sequencer is written in C++ causes the availability of the system, by simply recompiling on all the platforms supported by AMULET. At present these are a big set of UNIX/Xwindow flavors, Windows 95/NT and the MacOS.

In building the sequencer the mixed initiative model based on mutual constraining of behavior (Tate 1997) has been followed. The key point of this approach is the choice to share a common plan model between the users; on this model each user operate the desired manipulations. The system interprets each user action as an attempt to constraint the final aspect of the plan and check the feasibility of the proposed modification.

In the DRS system the user can be the commercial operator, interacting with the commercial-profile sequence, or the spacecraft engineer, interacting with the operative-profile sequencer. Each one of those users contributes to the final aspect of the plan by proposing the allocation of activities and by reacting to the changes due to the actions of the other users. All the users share a common, object-oriented vision of the scheduling domain that defines the scheduling vision and interact with the system via a personalized set of instruments. This view of the scheduling process is reported in Figure 6. The two scheduling clients, the commercial one and the operative one, are discussed in the following subsections.

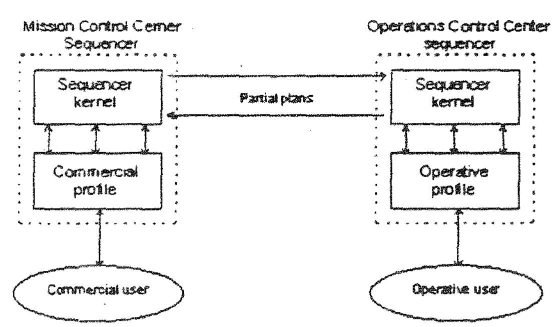

Figure 6: DRS O-OSCAR users views

The Commercial Profile

The commercial version of the sequencer is directed to the user of the Mission Control Center. The main screen consists of two windows, the main window and the Gantt chart window.

In the main window, Figure fig:comm, there are three lists with the information about the service requests proposed to the system, the requests accepted by the system and the requests refused by the system. The system allows loading and saving of partial schedules on the disk, the possibility to input a new request interactively via a dedicated mask and the capability to load a set of request from a disk file. The scheduling process consists on the selection of a request set and in the successive attempt to schedule it through the selection of the button Process. The interesting key points of this profile are:

- the attempt to abstract from the technical detail of the task;
- a simplified view of the scheduling task;
- exclusive tools to trade with the clients about the parameters of the requests refused by the system.

Regarding the last point this version of the sequencer provides the capability to select a set of requests from the refused requests list and the management for each request of the relative conflicting parameters. At this purpose is useful to use the Gantt chart to be able to directly the availability of the temporal parameters.

The Gantt window , Figure fig:gantt, allows the selection of each box associated with a request to have access to the relative detailed parameters. The successful trading of a request causes the apparition of the relative box in the Gantt chart.

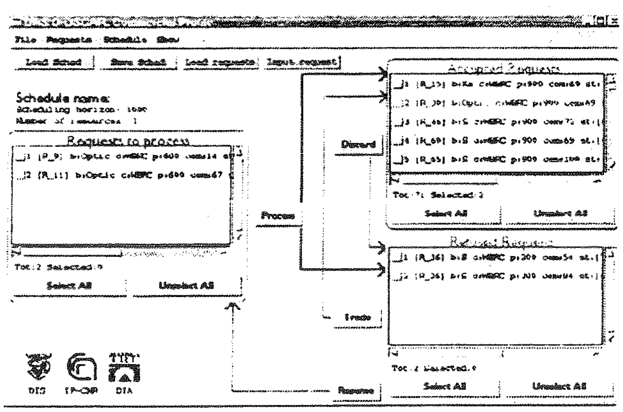

Figure 7: The interface for the commercial user

The Operative Profile

The operative version of the sequencer has the objective to incorporate in the partial plans created by the mission control center the requirements that allow

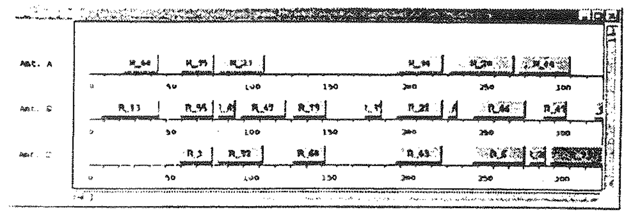

Figure 8: Current solution representation through a Gantt chart

the operativity of the orbiting platform (e.g. orbital shifts, maintenance of antennas), manage the available resources (e.g. excluding a broken antenna from the schedule) and allow the input of "last minute" requests (e.g. emergency operations). The need to minimize the effects of those operations claims full access to the available scheduling strategies provided from the system. When the program is started the screen shows the next window and the Gantt. The Gantt chart window is the same provided from the commercial profile sequencer.

In the main window of the Operative Profile, shown in Figure fig:oper the following information is continuously represented:

- The global state of the schedule.
- The total number of requests proposed to the system.
- The total number of rejected requests.

The details relatives to each of these aspects are accessible by the "More Info..." buttons. A dedicated button allows the use of the ESQ to evaluate the actual schedule quality (as explained below). The key points of this profile are:

- the possibility to alter the scheduling domain by removing an antenna;
- the ability to change the heuristic used to schedule the selected requests.
- the direct management of the backtracking.

As far as the last point is concerned the user has the capability to choose directly the activities to remove in case they are in conflict with the selected activity. In this way the modifications due to the backtracking are under the direct control of the user, allowing the minimum side effect of the reactive phase on the existent partial schedule.

System Performances

Two aspects of the system performance have been considered:

- the quality of the generated schedules;
- the time needed to generate the schedule.

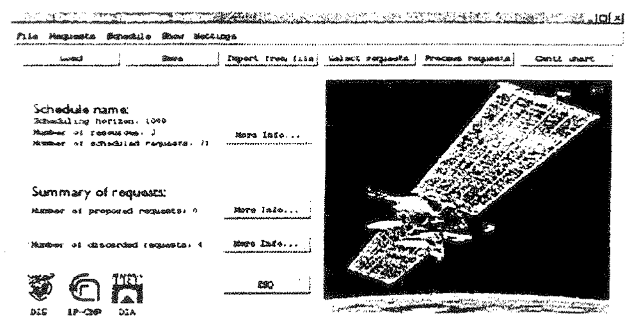

Figure 9: The interface for the operative user

As far as the first aspect is concerned, a version of the ESQ algorithm (Bresina *et al.* 1995) has been implemented which allows to compare the system solution with the average quality of randomly generated solutions.

This comparison is possible according to different quality criteria as shown in the Figure 10 in which the performance w.r.t. three aspects, flexibility, priority and global quality, are shown.

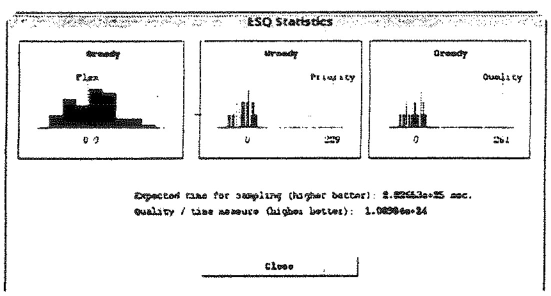

Figure 10: The ESQ evaluation of a solution

The system had be tested with the DRS problem simulator developed during a previous approach to the problem (Adinolfi and Cesta 1995). A typical daily DRS problem consists in 75 requests and it is integrally processed by the system in times compatible with the interactive use of the software. The 75 requests problem requires approximately 1 minute on the Windows 95 version of the sequencer with a Pentium at 75 MHz. The times on a typical UNIX workstation are reduced by one half. Figure 11 shows the CPU time (in seconds —y axis) needed to produce a solution for problems of increasing dimensions (number of requests —x axis) by using three different heuristics: the greedy algorithm, the greedy algorithm integrated with limited backtracking, and a general simulated annealing strategy used as a comparison. Figure shows timing on a Pentium 75MHz to stress the fact that the whole technological approach in portable also on widely available machines.

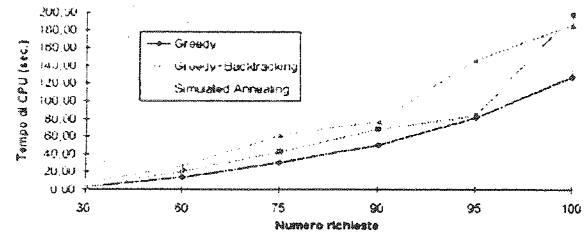

Figure 11: Performances on a Pentium 75MHz

Conclusions

In this paper we have described an approach to the scheduling of the DRS System requests based on a flexible and open architecture named O-OSCAR. The approach from one side relies on efficient algorithms for solution management and from another stresses the need for effective and personalized user interfaces.

The system obtains very interesting performances and solve several limitation of a previously developed rule-based approach (Adinolfi and Cesta 1995).

At present we are increasing the generality of the O-OSCAR framework developing a support for a more general domain description language, inspired by DDL.1 (Cesta and Oddi 1996a), and inserting the ability to deal with multiple capacity resource constraints as described in (Cesta and Stella 1997).

Acknowledgments

This research is supported by ASI — Italian Space Agency and is part of a joint project among Dipartimento di Informatica e Sistemistica dell'Università di Roma "La Sapienza", Dipartimento di Informatica e Automazione della Terza Università di Roma, and Reparto di Intelligenza Artificiale, Modelli Cognitivi ed Interazione at IP-CNR, Roma.

References

M. Adinolfi and A. Cesta. Heuristic Scheduling of the DRS Communication System. *Engineering Applications of Artificial Intelligence*, 8:147–156, 1995.

G. Booch. *Object-Oriented Analysis and Design with Application.* Benjamin Cummings, 1994.

J. Bresina, M. Drummond, and K. Swanson. Expected Solution Quality. In *Proceedings of IJCAI-95*, 1995.

R. Cervoni, A. Cesta, and A. Oddi. Managing Dynamic Temporal Constraint Networks. In *Artificial Intelligence Planning Systems: Proceedings of the Second International Conference (AIPS94)*, 1994.

A. Cesta and A. Oddi. DDL.1: A Formal Description of a Constraint Representation Language for Physical Domains. In M. M.Ghallab and A. Milani, editors, *New Directions in AI Planning.* IOS Press, 1996.

A. Cesta and A. Oddi. Gaining Efficiency and Flexibility in the Simple Temporal Problem. In *Proceedings of the Third International Workshop on Temporal Representation and Reasoning (TIME-96)*, 1996.

A. Cesta and C. Stella. A Time and Resource Problem for Planning Architectures. In *Proceedings of the Fourth European Conference on Planning (ECP 97)*, 1997.

ESA. DRS Preparatory Programme System Performance Specification, Issue 4, Rev. 1. Technical report, European Space Agency, ESTEC, 1989.

ESA. DRS Preparatory Programme Ground Segment Performance Specification, Issue 4, Rev. 1. Technical report, European Space Agency, ESTEC, 1990.

S. French. *Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop*. Ellis Horwood Lim., 1982.

B. Myers. *The Amulet V.2.0 Reference Manual*. http://www.cs.cmu.edu/~amulet.

A. Tate. Mixed-Initiative Interaction in O-Plan. In *Working notes of the AAAI Spring Symposium on Computational Models of Mixed-Initiative Interaction*, 1997.

G. Wolf. Schedule Management: An Object-Oriented Approach. *Decision Support Systems*, 11:373–388, 1994.